IN THE CLAIMS:

This listing of the claims replaces all prior versions and listings of the claims in this application.

The text of all pending claims (including any withdrawn claims) is set forth below. Canceled and not entered claims are indicated with claim number and status only. The claims as listed below show added text with underlining and deleted text with ~~strikethrough~~. The status of each claim is indicated with one of (Original), (Currently amended), (Canceled), (Withdrawn), (Previously presented), (New), and (Not entered).

Please AMEND claim 23 and ADD new claims 25-27 in accordance with the following:

1. (Previously presented) An apparatus for reproducing audio video (AV) data using a markup document in an interactive mode, comprising:

a buffer which buffers the markup document; and

a buffer manager which manages the buffer to preload the markup document and outputs buffering state information of the buffer in response to a report signal.

2. (Original) The apparatus of claim 1 further comprising a content decoder which interprets the markup document and outputs the report signal, wherein the buffer manager informs the content decoder of the buffering state information of the buffer in response to the report signal.

3. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using an application program interface (API).

4. (Original) The apparatus of claim 3, wherein the API serves to notify the content decoder of whether preloading of the markup document succeeded or failed, or whether the markup document is still being loaded.

5. (Original) The apparatus of claim 4, wherein the API returns a value of 0 where the preloading of the markup document succeeded, returns a value of 1 where the preloading of the

markup document failed, and returns a value of 2 where the markup document is still being loaded.

6. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using an API, which includes at least one of a file path and an attribute of the markup document as a parameter.

7. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using an [obj].isCached(URL, resType) API, where the URL is a parameter indicating a file path of the markup document and the resType is a parameter indicating an attribute of the markup document.

8. (Original) The apparatus of claim 2, wherein the buffer manager informs the content decoder of a buffering state of the markup document utilizing an API.

9. (Original) The apparatus of claim 1, further comprising a content decoder which interprets the markup document, wherein the buffer manager transfers the markup document from the buffer to the content decoder in response to a reproduce signal.

10. (Original) The apparatus of claim 1 further comprising a content decoder which interprets the markup document, wherein the content decoder outputs a release signal to the buffer manager indicating that the markup document therein brought from the buffer, in response to a reproduce signal, is not in use.

11. (Original) The apparatus of claim 10, wherein the content decoder outputs the release signal to the buffer manager in response to the markup document no longer being displayed in a screen of a display device.

12. (Original) The apparatus of claim 1 further comprising a content decoder which interprets the markup document, wherein the buffer manager deletes the markup document from the buffer in response to a discard signal output from the content decoder.

13. (Original) The apparatus of claim 12, wherein the content decoder generates the discard signal using a discard API.

14. (Previously presented) The apparatus of claim 2, wherein the content decoder generates the report signal using a progressNameOfFile API to determine a file name of the markup document currently being preloaded.

15. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using a progressLengthOfFile API to determine how much of the markup document currently being preloaded has been preloaded.

16. (Previously presented) The apparatus of claim 2, wherein the content decoder generates the report signal using a remainLengthOfFile API to determine how much of the markup document currently being preloaded is yet to be preloaded.

17. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using a totalLoadingSize API to determine a total load of the markup document to be preloaded.

18. (Original) The apparatus of claim 2, wherein the content decoder generates the report signal using a remainLoadingSize API to determine how much of a total load of the markup document is yet to be preloaded.

19. (Previously presented) An apparatus for controlling a buffer which buffers a markup document to reproduce audio video (AV) data in an interactive mode, comprising a buffer manager which manages the buffer to preload the markup document and outputs information of the buffer including buffering information of the markup document, wherein the buffering information includes:

information indicating that preloading of the markup document succeeded;

information indicating that the preloading of the markup document failed; and

information indicating that the preloading of the markup document is still be conducted.

20. (Original) The apparatus of claim 19, wherein the buffer manager outputs the information of the buffer using an application program interface.

21. (Original) The apparatus of claim 19, wherein the information of the buffer further includes information indicating whether a command to preload the markup document has been successfully received.

22. (Original) The apparatus of claim 19, wherein the information of the buffer further includes information indicating whether preloading of the markup document is completed.

23. (Currently amended) An apparatus for recording and/or reproducing audio video (AV) data using a markup document in an interactive mode, comprising:

an AV buffer which buffers the AV data;

an AV reproduction engine which decodes the AV data;

an enhanced ~~audio-video-~~navigation (ENAV) buffer which preloads the markup document to reproduce the AV data in the interactive mode;

an ENAV engine which identifies buffering state information of the markup document and decodes the markup document; and

means for obtaining the markup document.

24. (Original) The apparatus of claim 23, wherein the apparatus uses a blocked I/O method in response to obtaining the markup document from a data storage medium and an unblocked I/O method in response to obtaining the markup document from a network.

25. (New) The apparatus of claim 1, further comprising a reader which reads a preload-list file before the reproducing of the AV data begins;

wherein the buffer manager manages the buffer to preload the markup document based on contents of the preload-list file before the reproducing of the AV data begins.

26. (New) The apparatus of claim 25, wherein the preload-list file contains information identifying at least one markup document that is to be preloaded into the buffer under the control of the buffer manager before the reproducing of the AV data begins.

27. (New) The apparatus of claim 25, wherein the reader reads the preload-list file from an information storage medium.